人工神经网络的学习算法(1)

张鸿宾　（北京计算机学院）

摘要

人工神经网络可以看作是一种具有学习和自我组织能力的智能机器。最近几年，由于出现了一些强有力的学习算法，大大推动了有关神经网络的理论和应用研究。本文介绍并分析其中几个有代表性的学习算法。

关于智能机器的制造方法，很早以来一直有两种不同的策略在互相竞争着。第一种是以逻辑为基础的，认为只有符号运算才称得上是所谓智能的本质。这种策略是在通常的串行计算机上实现的，在专家系统和计算机博奕等领域中取得了很大的成功。但在处理和外部世界有较多联系的对象，诸如图像处理和声音、物体识别，以及常识推理等时，却显得能力不强，暴露出其局限性。

第二种策略是试图模仿人脑的信息处理机制，通过把大量的结构非常简单的计算单元（神经元）相互连接起来，构成人工神经网络，实现高度并行和分散的信息处理，用来进行物体和声音识别、联想存储和常识推理等。这种神经网络型计算机不需要编写程序，它可以从示例中进行学习，实现网络的自我组织。比起第一种策略来，这种策略在过去一些年进展不大，成果也不怎么显著。究其原因，主要有以下两点。一是过去的神经网络学习算法不够完善和强有力，信息表现的方式也不适当。二是在通常的串行机上模拟神经网络的效率非常低。然而，随着VLSI和CAD技术的进步，神经网络的硬件实现已经变得较为容易了。特别是由于近几年提出了一些强有力的学习算法，使得一度沉寂的神经网络型计算机的研究又出现了一个新的高潮。人们期待这个一度落后了的"小兄弟"能够成长起来，和"老大"——串行数字机携手，在未来的智能机器中发挥更大的作用。

本文在回顾以往的神经网络学习算法的基础上，介绍和分析最近几年新提出的两个主要学习算法——误差反向传播算法和Boltzmann Machine的学习算法，展望它们的

System Processor, Product Description, TRW MEAD AI Center, San Diego, 1987

[26] R. W. Kuczewski, et al, "Neurocomputer Workstations and Processors : Approaches and Applications", *IEEE ICNN Conf.*, 1987.

[27] J. Prichett, "The New Cybernetics : ADAPTing to AI", *TRW Messages*,4(1). 1986.

[28] L. D. Jackel,et al,"Electronic Neural Networks", in *Proc. of Supercomputing'88*, pp.41-46.

[29] L. D. Jackel, et al, "Electronic Neural Networks Chips", *Applied Optics*, 26, 1987.

[30] 蔡义发，丁德恒，吕维雪，电子神经计算机的设计与实现，《计算机学报》(待发表)。

一些可能应用并提出一些需要进一步探讨的问题。

一、神经网络的学习

神经网络的功能是由它的基本构成单元（以下称为神经元或网络的节点）的输入输出特性以及神经元之间连接的方式确定的。为了使网络能够执行一定的信息处理任务，我们必须规定网络的形态和各种参数，即规定神经元的输入输出特性（神经元的类型）、网络的连接方式、并设置各个神经元之间连接权的大小。

确定神经元之间连接权大小的方法目前主要有两种。一种是从所要解决的问题的定义中直接计算神经元之间的连接权。例如，当我们用Hopfield网络求解最优化问题时，可以令问题的评价函数和网络的能量函数一致，然后可以直接计算出各个连接权的大小。

另一种情况是，事先我们无法知道各个连接权的适当值，或者人为地设置各个连接权太繁琐的时候，必须使用"学习"的方法，或者称为网络的"自组织"方法。所谓神经网络的学习，就是通过给网络各种训练范例，把网络的实际响应和所希望的响应进行比较，然后根据偏差的情况修改各个连接权，使网络不断朝着能正确动作的方向变化下去，直到能得到正确的输出为止。如何根据偏差来确定各个连接权的修正量，这就是神经网络学习算法所要解决的问题。由于学习算法的优劣直接影响到神经网络求解问题的能力，所以有关神经网络学习算法的研究一直受到人们的重视。每当新的强有力的学习算法出现后，神经网络的理论和应用研究就向前迈进了一大步。

二、Perceptron和最小二乘的学习算法

Perceptron型网络是Rosenblatt在1958年提出的一种模式识别机[1]。它由S、A、R三层神经元组成。同一层内的神经元之间没有相互连接，不同层间也没有反馈。有学习能力的只是中间的A层和最后的R层之间的连接权。从这个意义上说，Perceptron实际上相当于一层有学习能力的神经网络（图1(a)）。最早的Perceptron的神经元是二值的，输入输出特性是阈值型的（图1(b)）。由于输出层的节点是互相独立的，所以只要考虑一个节点就可以了。R层的第j个节点所接受的总输入U_j^R和输出O_j^R分别为：

$$U_j^R=\sum_i W_{ij}\,O_i^A$$

$$O_j^R=f(U_j^R)$$

式中W_{ij}是A层的第i个节点和R层的第j个节点间连接的权。f是阈值型函数（图1(b)）：

$$f(x)=\begin{cases}1 & \text{当}x\geqslant 0\text{时}\\ 0 & \text{当}x<0\text{时}\end{cases}$$

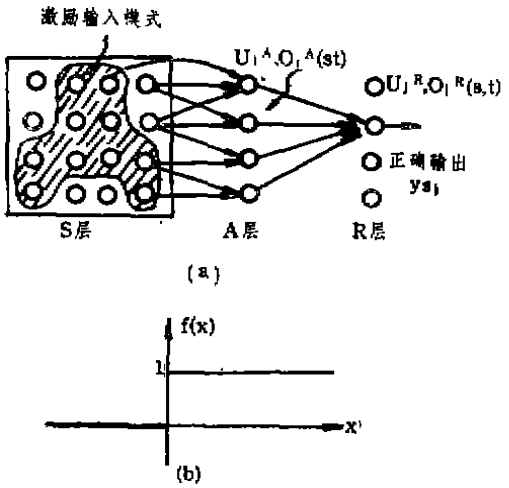

图1 Perceptron型网络及其输入输出特性

当把Perceptron型网络用作模式分类机时，在S层上给定某个输入模式（0，1的符号串）后，我们希望在R层上能够判定这个模式所属的类别。

这种网络的学习算法是[2]，先用各种模式对网络进行训练，当发生错误时，根据偏差来修改A层和R层之间连接权的大小。这是一种有教师的误差校正型学习方法。

设在t时刻的输入模式为$X_s=(x_{s1},\ x_{s2},\ \cdots\cdots,\ x_{sn})$，R层的神经元j对于$X_s$的正确输出设为$Y_{sj}$，把A层的神经元i的输出记为$O_i^A(s,\ t)$。R层神经元j的实际输出记为$O_j^R(s,\ t)$。那么，连接权$W_{ij}$可以按以下的方法加以修正：

$$W_{ij}(t+1)=W_{ij}(t)+(Y_{sj}-O_j^R(s,\ t))\cdot O_i^A(s,\ t)$$

式中Y_{sj}称为教师信号，$(Y_{sj}-O_j^R(s,\ t))$称为学习信号或误差信号。

对于上面的学习算法，可以证明下面的

收敛定理成立[4]：

定理1 当输入模式是线性可分时，上述学习算法可以在有限次数内收敛；这时所得到的各个连接权，能对所有的输入模式进行正确的分类。

上述的网络只能解决线性可分模式的分类问题。对于线性不可分问题，我们希望连接权能在误差的最小二乘意义下收敛。即网络的连接权能收敛到使下式的误差E达到最小：

$$E=\frac{1}{2}\sum_{s}\sum_{j}(Y_{s,j}-O^{R}_{j}(s))^{2}$$

只要对上面简单Perceptron网络稍加修改，把神经元由二值型改为连续取值，把输入输出特性由阶跃函数改为其它的单调递增函数，使用下面的连接权调整公式

$$W_{ij}(t+1)=W_{ij}(t)+\varepsilon(Y_{s,j}-O^{R}_{j}(s,\ t))\frac{dO^{R}_{j}(s,\ t)}{dU^{R}_{j}(s,\ t)}\cdot O^{A}_{i}(s,\ t)$$

就能保证网络的连接权近似地收敛到使E最小的值。式中dO^{R}_{j}/dU^{R}_{j}为神经元j的输出对输入的导数。ε为每步修正时的系数。

上述的学习算法实质上是连接权空间上的梯度下降法。当神经元的输入输出函数是单调时，定义在连接权空间上的误差曲面是椭圆抛物面，其水平断面是椭圆状，垂直断面是抛物线状，且最(极)小值只有一个。在这种情况下，梯度下降法是一种有效的方法。由于误差曲面的谷底一般较为平坦，可以预想算法的收敛速度将会很慢。必须小心地选择ε的大小以加快收敛速度并防止振动。

应当指出，Perceptron型网络的能力是有一定限度的。它们所能形成的判决区域只能由一个超平面来规定。有不少的问题，无论怎么调整连接的权，都不能实现输入和输出模式间正确的变换。例如，当我们试图用两个输入节点、一个输出节点的网络来作"异或"运算时，即当输入是(0，1)和(1，0)时输出1，当输入为(1，1)和(0，0)时输出0，这个网络无论怎样调节连接权和阈值，都不能胜任这个问题。对于Perceptron的能力和限度，Minsky和Papert在他们的书中作过相当透彻明晰的分析[3]。

为了提高神经网络的能力，可以把几层网络级联起来。关于多层网络的能力，可以证明有下面的完全性定理。

定理2 假定中间层节点可以根据需要自由设置，那么用三层的阈值型网络可以实现任意的二值逻辑函数。

定理3 假定中间层节点可以根据需要自由设置，那么用三层S状的输入输出特性的节点，可以以任意精度近似任何连续函数[4,5]。

把几层网络级联在一起时，中间层的神经元由于不直接和输入输出节点相连接，所以又叫作隐蔽节点。隐蔽节点可以用来学习输入模式各种有用的内部表示，大大提高了网络的功能。但是当人们试图把上面的误差校正型学习算法推广到多层网络上时，却遇到了很大的困难。由于有大量的和输出并无直接联系的隐蔽节点存在，当输出有误差时，如何判断这究竟是哪个神经元的"过错"呢？我们该调整哪个连接权呢？由于有这些难点，不少的人甚至怀疑对于多层网络来说存不存在这样的学习算法。直到1986年由Rumelhart、Hinton和Williams提出了多层网络的误差反向传播算法后(差不多同时Le Cun以及Parker也各自独立地提出了类似的算法)，多层网络的学习才有了重大进展，并随之出现了一大批引人注目的应用成果。下面一节我们以Rumelhart等人的算法为例[6]，说明多层网络的学习算法。

三、 多层网络的误差反向传播算法

这个算法的主要思路是，如果我们能求出误差E对各个神经元输出的偏导数，那么就可以算出E对所有的连接权的偏导数，因而可以利用梯度下降法来修改各个连接权。如果没有隐蔽节点，这很容易作到。因为要调整的连接权都和输出节点直接相连，E对各个输出节点的偏导数可以从E的定义中直接计

算出来。当有隐蔽节点存在时，问题就变得复杂了，但我们可以利用阶层型网络的特点，设法从输出层开始，反方向地、一层一层地把E对每个节点输出的偏导数求出来，从而确定E对各个连接权的偏导数。下面我们具体地分析一下网络的学习过程。

假定有一个M层的阶层型网络。把第K层的第i个节点记为U_i^K。U_i^K的输入是i_i^K，输出是O_i^K。输入输出函数记为f，$O_i^K=f(i_i^K)$。从U_i^K到U_j^{K+1}的连接权为$W_{i;j}^{k;k+1}$。设θ_i^K为U^K的阈值。θ_i^K的大小也可以学习。它的学习规则可以通过增设节点，转化为和连接权同样的学习方式。

对于输入模式P，如果输出层M的第j个节点的实际输出为O_j^M，正确的输出应该为Y_j的话，那么误差E_p可以定义为：

$$E_p=\frac{1}{2}\sum_j(Y_j-O_j^M)^2$$

为了减小误差E_p，根据梯度下降法，$W^{k-1;k}_{i;j}$的修正量$\Delta W^{k-1;k}_{i;j}$为：

$$\Delta W^{k-1;k}_{i;j}=-\varepsilon\frac{\partial E_p}{\partial W^{k-1;k}_{i;j}}$$

式中ε为修正系数。把上面的偏导数展开，有

$$\frac{\partial E_p}{\partial W^{k-1;k}_{i;j}}=\frac{\partial E_p}{\partial i_j^k}\cdot\frac{\partial i_j^k}{\partial W^{k-1;k}_{i;j}}=\frac{\partial E_p}{\partial i_j^k}\cdot O_i^{k-1}$$

当$k=m$时，由E_p的定义有

$$\frac{\partial E_p}{\partial i_j^k}=\frac{\partial E_p}{\partial O_j^m}\cdot\frac{\partial O_j^m}{\partial i_j^m}=-(Y_j-O_j^m)\cdot f'(i_j^m)$$

当$k\neq m$时，有

$$\frac{\partial E_p}{\partial i_i^k}=\sum_j\frac{\partial E_p}{\partial i_j^{k+1}}\cdot\frac{\partial i_j^{k+1}}{\partial O_j^k}\cdot\frac{\partial O_j^k}{\partial i_i^k}$$

$$=\sum_j\frac{\partial E_p}{\partial i_j^{k+1}}\cdot W_{i;j}^{k;k+1}\cdot f'(i_j^k)$$

$$=f'(i_j^k)\sum_j\frac{\partial E_p}{\partial i_j^{k+1}}\cdot W_{i;j}^{k;k+1}$$

把上面各式加以整理，可以得到连接权$W^{k-1;k}_{i;j}$的修正量的公式为：

$$\Delta W^{k-1;k}_{i;j}=-\varepsilon d_j^k O_i^{k-1}$$

式中$d_j^k=\partial E_p/\partial i_j^k$。当$k=m$时，

$$d_j^m=(O_j^M-Y_j)\cdot f'(i_j^m)$$

$k\neq m$时，$d_j^k=f'(i_j^k)\cdot\sum_l W_{j;l}^{k;k+1}\cdot d_l^{k+1}$

$(k=m=1,\ \cdots,\ 2)$

上式中的d_j^k可以看作是一般化的误差信号。它的计算是从$k=m$开始，到$k=2$递推地进行的。最后一式中的$\sum_l W_{j;l}^{k;k+1}$、d_l^{k+1}，就好象把在输出层所产生的误差信号d_j^m从输出层开始，以相反的方向，加权求和后一层层地传向输入层。这就是这种学习算法之所以叫作误差反向传播算法的由来。

如果网络所有节点的输入输出函数都是对数型的话，即

$$f(x)=1/(1+\exp(-x+\theta))$$

由于$f'(x)=f(x)(1-f(x))$，因此

$$f'(i_j^k)=O_j^k(1-O_j^k)$$

这是常常使用的一个公式。

在实际使用前面的学习算法时，为了加快收敛的速度，减小振动，可以采用如下的修正量公式：

$$\Delta W_{i;j}^{k-1;k}(t+1)=-\varepsilon d_j^k O_i^{k-1}+\alpha\Delta W_{i;j}^{k-1;k}(t)$$

式中的α为一小的正数，t表示学习的次数。

要注意的是，当网络含有隐蔽节点时，其误差曲面的极小值可能不只一个。和一般的梯度下降法一样，上述的学习算法也有可能收敛到某个极小值上。

下面我们以一个能检测模式对称性的网络为例，说明如何利用上面的学习算法进行网络的自我组织。这个例子非常有趣，网络的学习结果是我们事前所不曾料到的，仔细分析后又觉得网络学习的结果非常合理。

对于一个由0和1组成的模式串，我们希望用一个两层网络来判别它是否是中心对称的。例如，对一个六位长的模式串，当输入模式为010010或100001等中心对称模式时，网络的输出应为1。对010001、100100等非中心对称模式，网络的输出应为0。很明显，这是一个线性不可分问题。所使用的网络如图2所示，由6个输入节点、两个隐蔽节点和一个输出节点组成。每个节点是连续取值的。输入输出特性是对数型的。通过学习，我们希望找到一组适

当的权和节点阈值来解决这个问题。

把输入模式加在输入节点上后，使网络动作起来，计算实际输出和正确解之间的误差，然后利用上面的学习算法进行连接权的修正。连接权的初始值可以随机设置在－0.3到0.3不间。经过一千次左右的学习后，所得到的连接权和节点阈值如图2所示。

从图中的结果可以看出，所得到的连接权非常有趣。它们是左右镜象对称和上下镜象对称的，大小相等，符号相反。另外，连接权的大小之间有一个近似的1:2:4的关系。这使得输入节点的左3个或右3个节点的8种输入模式都以不同的值输入给隐蔽节点。当对称模式输入时，两个隐蔽节点的输出为0，输出节点的输出是1。当输入模式的中心对称性一旦破坏时，两个隐蔽节点之中至少有一个输出为1，因而网络的输出是0。上面的结果说明，通过学习，网络自身形成了一个非常有效的内部结构。这个内部结构的巧妙，甚至是我们事先所不曾料到的。

误差反向传播算法提出来以后，在模式识别、图像处理、数据压缩和人工智能等方面都取得了一批很有意义的研究成果。如Sejnowski和Rosenberg提出的把英文单词变换为发音记号的NET talk网络[7]，Sejnowski等人的识别海底岩石和潜水艇的声纳反射音的网络系统等，都取得了相当引人注

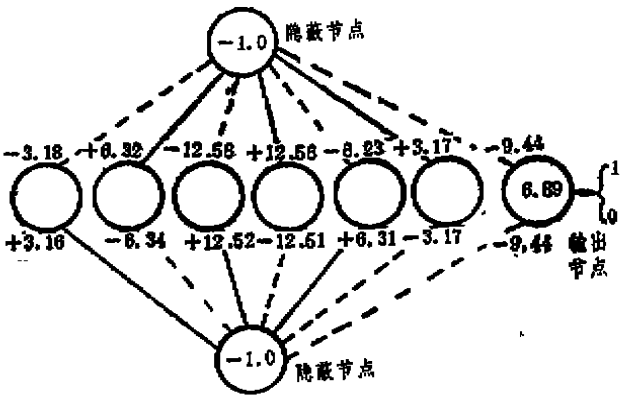

图2 求解中心对称的网络

目的成果。限于篇幅，这里就不准备介绍这个算法的一些实际应用了。

目前，误差反向传播算法的理论和应用还有不少的问题值得进一步研究。从收敛性上看，只要每步的修正量十分小的话，虽然可以收敛到误差平方和意义上的极小值点，但不能保证一定得到最小解。虽然在实践中有一些避开极小解的方法，但缺乏理论上的定量分析。从收敛速度上看，对小规模的问题还可以，对大规模的问题则相当慢。已经有人提出了一些加快收敛的方法[8,9]。另外，从少量的样本中学习的结果是否具有一般性等问题，也是传统模式识别理论中的一个难题。还有，从实际应用上看，中间层的隐蔽节点设置多少合适，如何设置等问题，也是值得研究的课题。目前是凭直感来进行的。有些实验说明，过多的中间层节点反而招致网络性能下降。所有以上这些问题，都有待于进一步的研究和实践。

参考文献

〔1〕 F. Rosenblatt, The perceptron, a probabilistic model for information storage and organization in the brain, Psychol. Rev., 65—6, 386—408, 1958.

〔2〕 F. Rosenblatt, Principles of Neurodynamics—Perceptrons and the Theory of Brain Mechanisms, 1961.

〔3〕 M. Minsky, and S. Parpert, Perceptrons, 1969.

〔4〕 B. Irie, and S. Miyake, Capabilities of three-layered Perceptrons, Proc. of IEEE ICNN88 1988.

〔5〕 船橋賢一，ニューラルネットワークのCapability について (On the Capability of reural network)，(日本)電子情報通信学会技術报告，MBE88—52，1988.

〔6〕 D. E. Rumelhart, et al, Learning internal representations by error propagation, In Parallel Distributed Processing, Vol.1, 1986.

〔7〕 T. J. Sejnowski, and C. R. Rosenberg, Parallel networks that learn to pronounce English text, Complex System, 1, 145—168, 1987

〔8〕 木本隆，ほか，ニューラル・ネットワークの学習方式の検讨，昭和63年度人工智能学会全国大会论文集，pp.123—124，1988

〔9〕 L. P. Ricotti, S. Ragazzini, anp G. Martenelli, Learning of word stress in a sub-optimal second order back-propagation neural network, Proc. of IEEE ICNN 88, 1988.